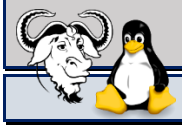


PRACTICAL WORK N°4

Files, Directories and Permissions Management

➲ A Reminder !

⌦ **Terminology:**

File ; Directory ; Absolute Path ; Relative Path ; Symbolic Link ; Physical Link ; inodes ; Permissions ; ...

⌦ **The file management system (FMS)**

- **Filesystem Hierarchy Standard (FHS) :** The file system starts from the **root directory /** and branches out into various **subdirectories**. Some important directories include:
  - /bin - Essential binary executables
  - /etc - System configuration files
  - /home - User home directories
  - /var - Variable data (logs, caches, etc.)
  - /tmp - Temporary files
  - /dev - Device files (representing hardware devices)
  - /mnt or /media - Mount points for external devices
  - ...

- **Types of File Systems** : **ext4** (*Fourth Extended File System*), **Btrfs**, **XFS**, **FAT32**, **NTFS** ;
- **Mounting and Unmounting File Systems** (Commands such as : **mount**, **unmount**);
- **File Permissions and Ownership :**
  - Each file has three types of permissions: Read **(r),** Write **(w),** Execute **(x)**;
  - User and Group Ownership: The owner and the group have specific permissions.
  - Commands such as **chmod**, **chown**, and **chgrp** are used to manage file permissions and ownership.

- **File Types :**
  - **Regular files:** Contain data (text, binary, etc.).
  - **Directories:** Contain other files or directories.
  - **Symbolic links:** Pointers to other files or directories.
  - **Special files:**
    - ✓ ***Device files****: Represent hardware devices (e.g., /dev/sda).*
    - ✓ ***Named pipes and sockets****: Used for inter-process communication.*
- **File Operations and Tools :** Linux provides a wide array of commands for file manipulation.
- **File System Utilities :** Linux provides various tools for managing file systems and disk partitions :
  - **fdisk** / **parted** – For partitioning disks.
  - **mkfs** – For creating file systems.
  - **fsck** – For checking and repairing file systems.
  - **resize2fs** – For resizing ext file systems.
- ...

⌦ **Files, Directories:**

- **Special directories:**
  . represents the current directory,
  .. represents the parent directory
  ~ represents the user's home directory
- **Hidden files:** On UNIX, hidden files start with a period. For example, **~/.bashrc** is a hidden file, in the user's home directory, that contains the configuration of his shell.
- **Wildcards: ?** and ***** The characters **?** and ***** in file and directory names are used to represent any character. '**?**' represents a single character, while '*****' represents any number.

⌦ **The SHELL :**

There are several types of shells, the most well-known since UNIX having an improved version on Linux. The **/etc/shells** file contains a list of all available shells (for example: **/bin/bash**).

Basic Commands :

- Files and Directories Management : ...
- Permissions Management : ...

Useful Command Shortcuts :

These shortcuts can help you navigate the terminal more efficiently.

- Ctrl + C**: Cancel the current running process.**
- Ctrl + D**: Exit the current shell (**logout**).**
- Ctrl + Z**: Suspend the current process (**send it to the background**).**
- Ctrl + A**: Move the cursor to the beginning of the line.**
- Ctrl + E**: Move the cursor to the end of the line.**
- Ctrl + U**: Delete everything before the cursor.**
- Ctrl + K**: Delete everything after the cursor.**
- Ctrl + W**: Delete the word before the cursor.**
- Ctrl + L**: Clear the terminal screen (**equivalent to the clear command**).**
- Ctrl + R**: Search through the command history (**press repeatedly to scroll through older commands**).**
- Ctrl + T**: Swap the characters before and after the cursor.**
- Alt + F**: Move the cursor forward one word.**
- Alt + B**: Move the cursor backward one word.**
- Arrow Up/Down**: Scroll through the command history.**
- Tab**: Auto-complete a command, filename, or directory.**
- **!! – Repeat the last command.**
- **...**

Useful Shell Aliases:

An **alias** is a shortcut for a longer command or sequence of commands. Aliases can be defined in the **shell's configuration files** (like **.bashrc** or **.zshrc**) for persistent use.

➲ Notes & Practical Tips

Connection : To work on a **Linux** distribution, you need to have an account, so enter your username (login) and password.

- **Username**: 1BTg**X**b**Y** (**X** is your **Group** number and **Y** is your **Machine** number)
- **Password**: NSCS-s**Z** (**Z** is your section number)

Attention :
UNIX/Linux differentiates between uppercase and lowercase. Most commands should be written in lowercase. Always separate the command from its arguments with spaces.

➲ Objectives

- Learn about the file system;
- Learn about absolute and relative paths;
- Learn and master the basic commands for managing files and directories.
  - ✓ Move around the system tree ;
  - ✓ Create a text file, display its contents and modify it;
  - ✓ Managing Permissions;
  - ✓ Use the most useful Linux commands;
  - ✓ ...

PART 1: THEORETICAL PART / COURSE QUESTIONS

1. **The FMS (File Management System) is** a system that controls the storage, retrieval, and organization of files on a storage device; **its role is** to manage files, directories, file permissions, and file systems, ensuring data integrity, security, and efficient access across the operating system.
2. **The basic elements of a file system are**: files, directories, file types, file metadata (such as *size*, *timestamps*, *permissions*), storage devices, file system structure (such as ***inodes***), and file management utilities.
3. **On Linux, the filesystem structure is of the form**: a hierarchical tree structure; its **vertex** is called: the **root directory**, represented by **/**.
4. **Is your Desktop a file or directory ?** : the "**Desktop**" is a **directory** where files and shortcuts are stored.
5. **A path is**: a string that specifies the location of a file or directory within the file system; There are **two types**: **absolute path** (which starts from the **root /**), and **relative path** (which is relative to the current working directory).

File and Directory Management	
cp	Copies files and directories
pwd	Prints the current working directory
cd	Changes current directory
cd..	Changes to the parent directory
cd -	Changes to the previous directory
cd ~	Changes to the home directory
ls	Lists contents of the current directory
ls -l	Lists the files and directories in the current working directory in long format
ls -a	Lists all files and directories including the hidden ones
ls -r	Lists files/directories in reverse
mkdir	Creates a new directory
rmdir	Removes an empty directory
mv	Moves or renames a file /directory
rm	Removes a file/directory
rm -i	Prompts system confirmation before deleting
rm -r	Deletes a file/directory recursively
rm -d	Deletes empty folders

PART 2 : ACTIVITIES

✪ ACTIVITY 1 : *Location Commands (Navigate the file system tree)*

pwd :

- Signification : **p**---------- **w**---------- **d** ----------
- Displays the current directory so-called *working directory*.

Example of use : `$ pwd`

What the type of the path; absolute or relative? : ----------

cd :

- Signification : **c**---------- **d**----------
- Allows you to browse through directories

Examples of use : `$ cd` *Allows you to return to the /home/user directory (same as* `$cd ~` *)*

`$cd -` *Allows you to return to the* ***previous directory***

`$cd ..` *Allows you to go back to the* ***parent directory***

`$cd /` Allows you to go back to the ----------

`$cd /usr/share/doc/` Allows you to place yourself in the directory ----------

tree :

1. *Allows you to display the file system tree from the directory passed as an argument (the default current directory).*
2. *Without arguments, then displays the entire tree from the current directory, which is represented by a .*

✪ ACTIVITY 2 : *Consultation Commands*

ls :

- Signification : **l** ----------
- Allows you to list a directory

Note: You can use several options at the same time as a sequence : `$command option1 option2 ...`

Examples of use: `$ ls -l -a` or `$ ls -la` or `$ ls -l --all`

Some options of the command **ls** :

-l : Allows for a display ----------

-h : Allows ----------

-a : Allows the display of files and directories ---------- those that start with a **.** (**dot**))

-lct : Allows you to ---------- files and directories by ---------- of decreasing changes.

Examples of use :

`$ ls -a` Displays all ---------- in the current directory.

`$ ls /etc/` Displays the contents of the directory ----------

cat :

- Signification : ----------

- Displays the contents of a file

Some options for use:

-n : ----------

-v : ----------

Example of use: **`$cat -n /etc/passwd`** *Displays **passwd** by numbering the lines from 1*

more :

- Signification : ----------

- Displays one file page by page

Some options for use :

-s : ----------

-f : ----------

Example of use : **`$more -sf /etc/passwd`**

*Displays the **passwd** file container page by page, concatenating empty lines without cutting long lines.*

less :

- Signification : ----------

- Displays a file by allowing navigation.

Some options for use :

-e or **-E** : Automatically exits the second time the end of the file is reached, or the first time with **-E**.

-F : ----------

-m or **-M** : long prompt to the command **more**.

-r or **-R** : Allows special characters.

-x : Adjusts tab size.

-~ : does not fill in empty lines with **~**

Example of use : **`$ less -Emr~ /etc/passwd`**

*Displays **passwd** page by page with a long prompt (display the percentage of the file browsed) displaying special characters without filling in empty lines with **~**.*

✪ ACTIVITY 3 : *Manipulation Commands*

mkdir :

- Signification : **m** ---------- **d** ----------

- Creates an empty directory

Some option of the command **mkdir** :

-p : ----------

Example of use :

`$ mkdir TP_LINUX` *Creates the directory TP_LINUX*

touch :

- Signification : ----------

- Used to change the dates of access and modification of a file or to create a new one.

Some options for use :

-a : ----------

-m : ----------

Examples of use :

```
$ touch test.txt
```

touch allows you to change the dates of access and modification of a file test.txt if it exists, if not, you create it again.

mv :

- Signification : .....

- Allows you to move or rename files and directories.

Some option of the command **mv :**

-f : ----------

-i : ----------

-u : ----------

Examples of use :

```
$ mv file Documents/              Moves file to the directory Documents
$ mv Documents/file Desktop/      Moves the file file from the directory Documents to the Desktop
$ mv Documents Documents2         Renames the directory Documents to the directory Documents2
$ mv file file2                   Renamed file file to file file2
```

cp :

- Signification : .....

- Allows you to copy files or directories

Some option of the command **cp** :

-a : ----------

-i : ----------

-f : ----------

-r : ----------

-u : ----------

-v : ----------

Examples of use :

```
$ cp file Documents/              Copies the file file to the directory Documents/
$ cp -r Desktop/ Documents/       Copies the directory Desktop to the directory Documents/
```

rmdir :

- Signification : **r** ---------- **d** ----------

- Deletes a directory (empty)

Some options for use :

-p : ----------

Examples of use : `$ rmdir TP_LINUX` *Deletes the TP_LINUX directory*

rm :

- Signification : ----------

- Allows you to delete files

Some options of the command **rm** :

-f : ----------

-r : ----------

Examples of use :

```
$ rm file          Delete the file
$ rm -rf /home/user/TP_LINUX
```

Deletes the directory /home/user/TP_LINUX and all its files without asking for confirmation.

✪ ACTIVITY 4 : *Manipulating the contents of a file*

1. To create an (empty) file, you can use the touch command. Create a file named ***awesomefile.txt***.
   Solution : `touch awesomefile.txt`
2. The command **echo** allows you to write to the standard output. Try it with the *NSCS school* parameter.
   Solution : `echo NSCS school`
3. You can also redirect the output of **echo** *NSCS school* to a file, which allows you to write to it. **>** is used for this redirection. It overwrites the contents of the file; If you just want to add content to the end of the file use **>>**
   Solution : `echo NSCS school > awesomefile.txt`
4. Check the result of the previous question using the cat command.
   Solution : `cat awesomefile.txt`
5. **echo** is not the only command whose output can be redirected. Write the manual for the command **cat**, in ***awesomefile.txt***.
   Solution : `man cat > awesomefile.txt`
6. **For** files larger than a certain size, the command **cat** is impractical. Check the result of the previous question using **less**.
   Solution : `less awesomefile.txt`

✪ ACTIVITY 5 : *File Manipulation*

1. Using the command **cp** (*copy*), copy ***awesomefile.txt*** to your home. Check the result with **ls**.
   Solution : `cp awesomefile.txt ../copied.txt && ls ..`
2. The command **mv** (*move*) allows you to move/rename files. Rename the file you just copied to your home, and don't forget to check the manual.
   Solution : `mv ../awesomefile.txt ../newName.txt`
3. Now delete this file, using the command **rm** (*remove*).
   Solution : `rm ../newName.txt`
4. What would have had to be changed in the previous questions if we had handled a **directory** (empty/non-empty)?
   Solution :
   `cp` : add `-r`
   `mv` : nothing to add
   `rm` : add `-r,` or use `rmdir` if empty

✪ ACTIVITY 6 : *File and Directory Management : Absolute & Relative paths*

1. Run the command **pwd** and interpret the result:

   ---

2. Move to the root directory (**/**), the command used is:

   ---

3. Then move to the directory (**/tmp**) , the command used is:

   ---

4. From the current directory (**/tmp**) indicate the path to your working directory in two different ways:

a. Using an **absolute path :**----------

b. Using a **relative path :**----------

5. Using either the absolute or relative path to access your working directory, the command used is:

---

6. In your working directory, create a directory named **LiNuX** and then move to that directory.

---

7. Create a directory named **UbuntuOS** and then move yourself to that directory**.**

---

8. What should the **pwd** command display ? ---------- Run the command to confirm.
9. What is the effect of running the command **cd ..** ? How to check the result ?

---

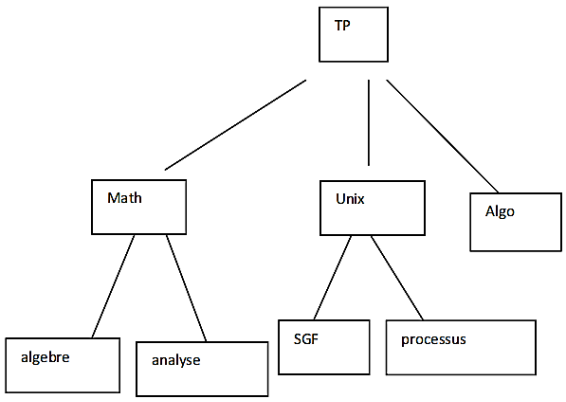
10. Create a directory named **NSCS,** the command used is: ----------
11. Again, run the command **cd ..** in which directory do you position yourself ? how to confirm?

---

12. Draw the tree created by the previous commands?

✪ ACTIVITY 7 : *File and Directory Management*

Create the following **tree** in your working directory.



A. Specify the path to the directory **SGF** from your working directory

---

B. Specify the path to the directory **SGF** from the directory **TP**

C. Specify the path to the directory **algebre** from the directory **Unix**.

D. Specify the path to the directory **algebre** from the directory **Math**.

E. Specify the path to the directory **SGF** from the directory **analyse**

F. Specify the path to the directory **SGF** from the **Algo** directory

---

G. Specify the path to the directory **Algo** from the directory **analyse**.

H. Specify the ***absolute path*** to the directory **algebre**.

I. Specify the ***absolute path*** to the directory **Algo**.

---

1. Run the command **(ls**) with and without the option **(–l**) in your working directory. What is the difference between the two displays?

---

2. View the list of hidden files. The command used is:

---

3. Move yourself to the **/usr/include** directory and view all the files in that directory with their characteristics. Name the useful commands:

---

4. Then run **ls -l *.h** What is the difference between this display and the display of the previous question?

---

5. Then, run **ls -l z*.h** What is the difference between this display and the display of the previous question?

---

6. Run **ls -l *e.h** What is the difference between this display and the display of question **4**?

---

7. Then run **ls -l ???. h** What is the difference between this display and the display of question **4**?

---

8. Learn about the following options of the command **ls**: **-ld , -lt**

---

9. Return to your working directory; The command is: ----------

10. Move to the directory **LiNuX** and create two new empty files **new1** and **new2**

a. ----------

b. ----------

11. Copy the file **new1** to the directory **UbuntuOS.**

---

12. Move the file **new2** to the directory **OS1**.

---

13. Delete the file **new1** from the directory **LiNuX**

---

14. In *your working directory,* Run the command **mkdir –p rep1/rep2/rep3** ; What is the result of this command

---

---

15. Move to the directory **UbuntuOS,** confirm with the command **pwd.** Copy the file **new1** to the **rep3** , **rep2** and **rep1** directories by changing its name **(tp2unix**):

a. ---

b. ---

c. ---

16. Copy the directory **rep3** into the directory **UbuntuOS.**

---

17. Move the directory **rep2** to the directory **OS1.**

---

18. Rename the directory **rep1** to **Folder1**

---

19. Try to delete the directory **rep2** (in the **OS1**) with the **rmdir** . it doesn't work, why? find the command to use?

---

20. Finally, draw the **tree diagram** after the execution of the previous commands?

✪ ACTIVITY 8 : *Manipulating Files and Directories*

Let us consider the following commands:

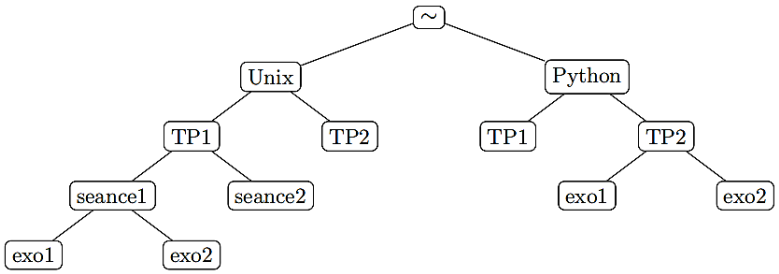
1. `mkdir -p Banque/Agence1 Banque/Agence3`
2. `cd Banque`
3. `mkdir -p ./Agence2/Rep1`
4. `touch Agence1/Client1 Agence2/Client2 Agence3/Client3`
5. `cd ./Agence2/Rep1`
6. `cp ../../Agence1/Client1 ../../Agence3/Client6`
7. `cp ../Client2 ../../Agence1/Client4`
8. `touch ../Client5`
9. `mv ../../Agence1/Client1`

Draw the tree obtained from the previous commands.

✪ ACTIVITY 9 : *Manipulating Files and Directories*

1. Create the files in your home directory according to the following tree:

   And in the following way, each question must be answered in a single command and the current directory must always be your personal directory:



   a. Create two directories : **Unix** and **Python** in your home directory.
   b. Create two directories : **TP1** and **TP2**, in the **Unix** directory.
   c. Copy the both directories **TP1** and **TP2** into the **Python** directory.
   d. Create two directories : **session1** and **session2** in the directory **TP1** in **Unix**.
   e. Create two files **exo1** and **exo2** in the directory **session1**.
   f. Copy the two files **exo1** and **exo2** to the directory **TP2** in **Python**.

2. Move to the directory **Python/TP1**. Make sure you're there, and then, with a single command, move to the directory **Unix/TP2** using the ***relative path*** of the directory. Go back to the directory **Python/TP1**, but this time, using the ***absolute path***. Finally, go back to the root of your home directory.

3. Remove the directory **TP1** from **Python** with the command **rmdir**.

4. Remove the directory **TP2** from **Python** with the command **rmdir**, why the command failed. Delete the directory **TP2** from **Python** in one command.
5. Move the file **exo1** to the directory **TP1** and delete the directories **session1** and **session2**.
6. Rename the files **exo1** and **exo2** to **exercise1** and **exercise2** respectively.
7. Draw the **tree structure of the files** contained in the home directory in a theoretical way. Check it with a command.

SOLUTION

1.

a. `> mkdir Unix Python`

b. `> mkdir Unix/TP1 Unix/TP2`

c. `> cp -R Unix/TP1 Unix/TP2 Python`

d. `> mkdir Unix/TP1/seance1 Unix/TP1/seance2`

e. `> touch Unix/TP1/seance1/exo1 Unix/TP1/seance1/exo2`

f. `> cp Unix/TP1/seance1/exo1 Unix/TP1/seance1/exo2 Python/TP2`

2.
```
> cd Python/TP1
> pwd
> cd ../../Unix/TP2
> cd nom_du_chemin_absolu/Python/TP1
> cd
```

3. `> rmdir Python/TP1`

4. `> rmdir Python/TP2`

   The command fails because the directory **TP2** is not empty. `> rm -R TP2`

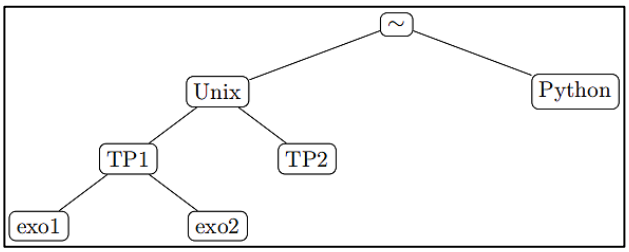
5. `> mv Unix/TP1/seance1/exo1 Unix/TP1/exo1`

   `> rm -R Unix/TP1/seance1 Unix/TP1/seance2`

6. `> mv Unix/TP1/exo1 Unix/TP1/exercice1`

   `> mv Unix/TP1/exo2 Unix/TP1/exercice2`
   `> ls -R`

7. **The Tree structure :**



✪ ACTIVITY 10 : *Manipulating Files and Directories*

By analysing this scenario, fill in the missing parts: name of the command, options, parameters or results of command execution so that the tasks requested can be carried out correctly.

The commands must be executed in order and the execution of a command depends on the execution of previous commands.

The execution of a command will therefore be applied to the results of previous commands.

```
admin1@PC:~/NSCS$ tree -F  ..
..
├── Bureau/
│   └── CR
└── NSCS/
    ├── CP/
    └── CS/

4 directories, 1 file
```

1	Create the files **1cp** in **CP** and **cf** in **Bureau**	admin1@PC:~/NSCS$ touch CP/ ./../ **admin1@PC:~/NSCS$ touch CP/1cp ./../Bureau/cf**
2	Rename the file **CR** to **cr**	admin1@PC:~/NSCS$ mv ../ ../ **admin1@PC:~/NSCS$ mv ../Bureau/CR ../Bureau/cr**
3	Move **NSCS** to **Bureau**	admin1@PC:~$ mv ./ ./ **admin1@PC:~$ mv ./ESI/ ./Bureau/**
4	Create the directories: "**SYS 1"** in **Bureau** and the "**2023 Exam"** in the directory "**SYS 1"**	admin1@PC:~$ mkdir Bureau/ **admin1@PC:~$ mkdir -p Bureau/"SYS 1"/"Examen 2023**
5	Delete content from the desktop except **NSCS**	admin1@PC:~$ rm Bureau/ **admin1@PC:~$ rm -r Bureau/!(NSCS)**
6	Give only directory size **NSCS** in human format	admin1@PC:~$ du Bureau/NSCS **admin1@PC:~$ du -s -h Bureau/NSCS**
7	Search from the home directory on files or directories that have the name **CS**	admin1@PC:~$ find **admin1@PC:~$ find ~ -name "CS"**
8	Know the type of **cd**	admin1@PC:~$ cd is a shell builtin **admin1@PC:~$ type cd** cd is a shell built-in
9	Delete **1cp**	admin1@PC:~$ **admin1@PC:~$ rm Bureau/NSCS/CP/1cp**
10	Give the result of the command.	admin1@PC:~$ echo Bureau/*/* **admin1@PC:~$ echo Bureau/*/*** **Bureau/NSCS/CP Bureau/NSCS/CS**